SERVOMOTORES

SERVOMOTORES

- Un servo es un tipo de accionador ampliamente empleado en electrónica.
- En un servo indicamos directamente el ángulo deseado y el servo se encarga de posicionares en este ángulo.
- Típicamente los servos disponen de un rango de movimiento de entre 0 a 180°.
- No son capaces de dar la vuelta por completo (de hecho disponen de topes internos que limitan el rango de movimiento).
- Internamente un servo frecuentemente consta de un mecanismo reductor.
- Proporcionan un alto par y un alto grado de precisión (incluso décimas de grado).
- Las velocidades de giro son pequeñas frente a los motores de corriente continua.

SERVOMOTORES

- Los servos admiten una tensión de alimentación entre 4,8V a 7,2V, siendo el valor más adecuado es 6V.
- Con tensiones inferiores el motor tiene menos fuerza y velocidad.
- Con tensiones superiores a 6,5V los servos empiezan a oscilar demasiado, lo cual los hace poco útiles.
- Los servos son cómodos de emplear, ya que ellos mismos realizan el control de posición.
- Ampliamente empleados en proyectos de robótica, como brazos robóticos, robots con patas, controlar el giro de torretas, u orientar sensores.

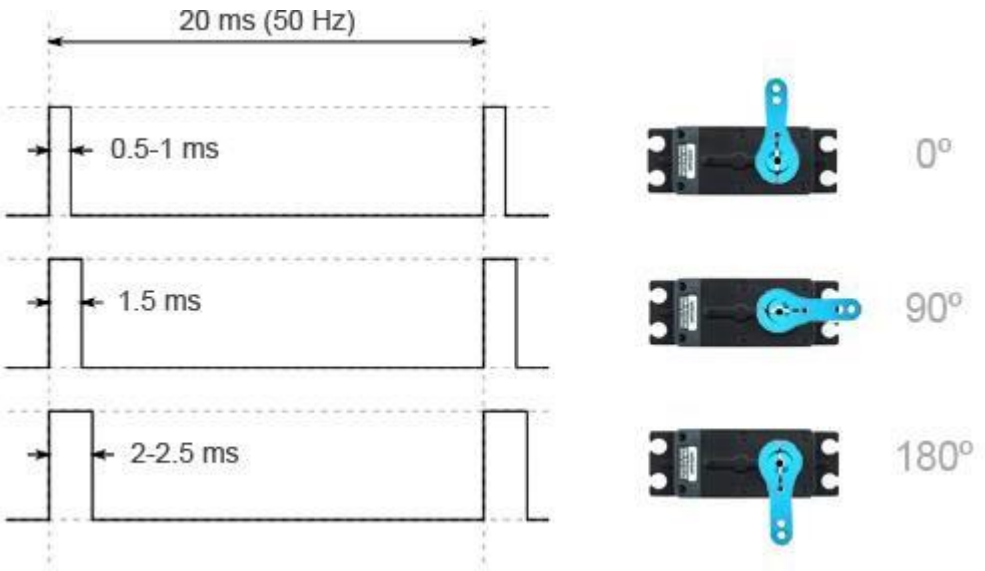
SERVOMOTORES. CÓMO FUNCIONA UN SERVO?

- Internamente un servo está constituido por un motor de corriente continua, acoplado a un reductor para reducir la velocidad de giro, junto con la electrónica necesaria para controlar su posición.

SERVOMOTORES. CÓMO FUNCIONA UN SERVO?

- Frecuentemente se dispone de un potenciómetro unido al eje del servo, que permite al servo para conocer la posición del eje.
- Esta información es tratada por un controlador que se encarga de ajustar y actuar sobre el motor para alcanzar la posición deseada.

- La comunicación de la posición deseada se realiza mediante la transmisión de un señal pulsada con periodo de 20ms. El ancho del pulso determina la posición del servo.



SERVOMOTORES. CÓMO FUNCIONA UN SERVO?

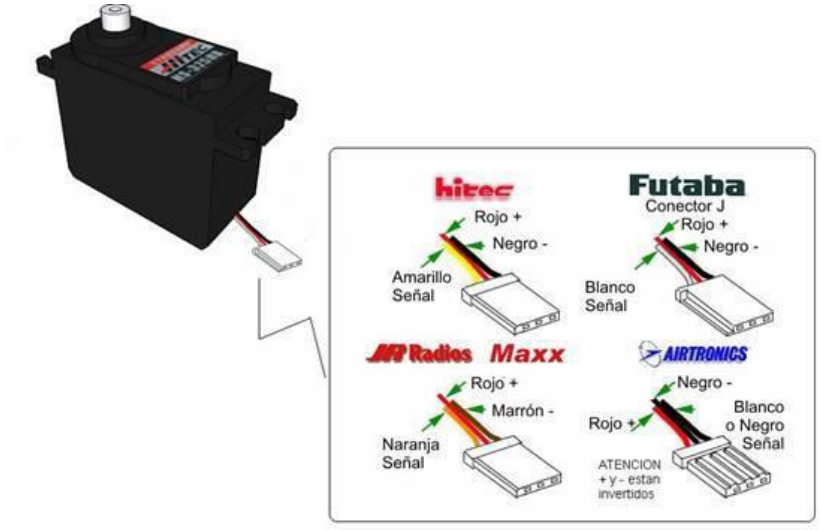
- La relación entre el ancho del pulso y el ángulo depende del modelo del motor.
- Algunos modelos responden con 0° a un pulso de 500 ms, y otros a un pulso de 1000 ms.

- En general, en todos los modelos:
  - Un pulso entre 500-1000 us corresponde con 0°
  - Un pulso de 1500 ms corresponde con 90° (punto neutro)
  - Un pulso entre 2000-2500us corresponde con 180°

- Por tanto, variando la señal en microsegundos podemos disponer de una precisión teórica de 0.18-0.36°, siempre que la mecánica del servo acompañe.

SERVOMOTORES. ESQUEMA DE MONTAJE

- Conectar un servo a Arduino es sencillo. El servo dispone de tres cables, dos de alimentación (GND y Vcc) y uno de señal (Sig).
- El color de estos cables suele tener dos combinaciones:
  - Marrón (GND), Rojo (Vcc) y Naranja (Sig)
  - Negro (GND), Rojo (Vcc) y Blanco (Sig)
- Por un lado, alimentamos el servo mediante el terminal GND (Marrón/Negro) y Vcc (Rojo).



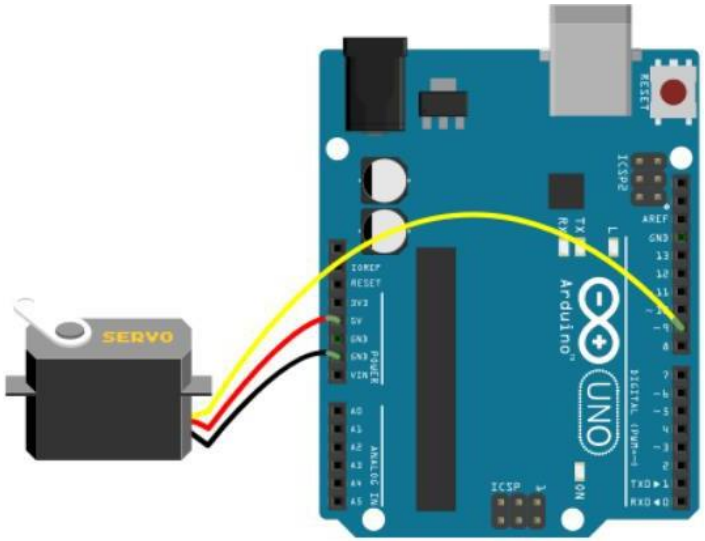
SERVOMOTORES. ESQUEMA DE MONTAJE

- En general, la alimentación a los servos se realizará desde una fuente de tensión externa (una batería o fuente de alimentación) a una tensión de 5V-6.5V, siendo 6V la tensión idónea.
- Arduino puede llegar a proporcionar corriente suficiente para encender un servo pequeño (SG90), suficiente para hacer unos cuantos proyectos de prueba.
- Sin embargo no dispone de corriente suficiente para actuar un servo grande (MG996R). Incluso varios servos pequeños, o hacer excesiva fuerza con ellos puede exceder la capacidad de corriente de Arduino, provocando su reinicio.
- Por otro lado, finalmente, para el control conectamos el cable de señal (naranja / blanco) a cualquier pin digital de Arduino.

SERVOMOTORES. EJERCICIOS

- El control de servos en Arduino es muy sencillo, ya que el IDE Standard proporciona la librería "servo.h", que permite controlar simultáneamente hasta 12 servos en Arduino Uno/Nano y hasta 48 servos en Arduino Mega.
- Entre los ejemplos típicos para ilustrar el funcionamiento de servos tenemos el Sketch "Sweep", que realiza un barrido continuo con el servo.

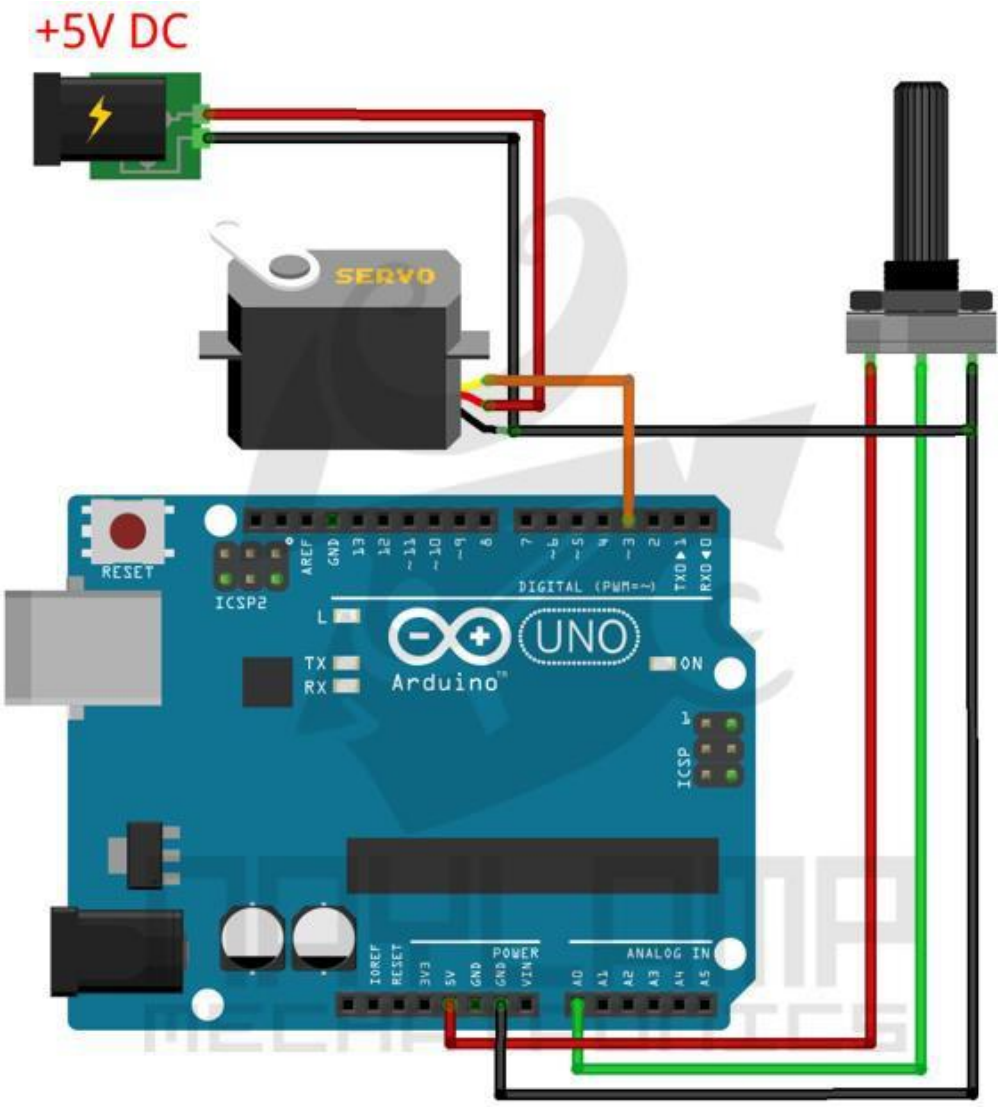
- Para ello incrementa el ángulo de 0 a 180° a razón de 1° cada 15ms, posteriormente realiza la operación contraria de 180° a 0°, para finalmente reiniciar el bucle.



SERVOMOTORES. EJERCICIO 1

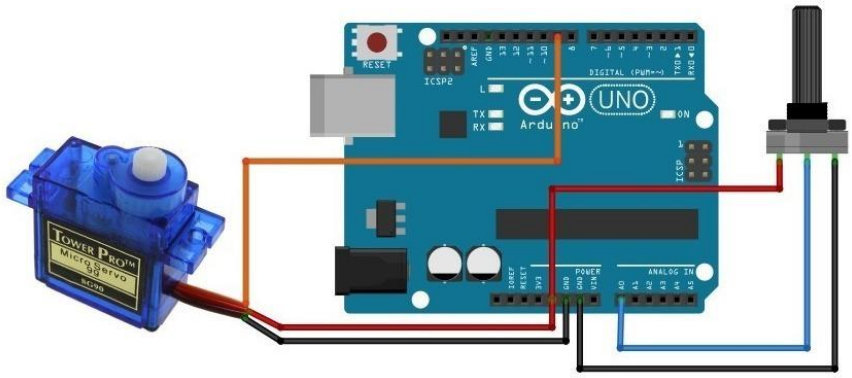
```
#include <Servo.h>

Servo myservo;  // crea el objeto servo

int pos = 0;    // posicion del servo

void setup() {
   myservo.attach(9);  // vincula el servo al pin digital 9
}

void loop() {
   //varia la posicion de 0 a 180, con esperas de 15ms
   for (pos = 0; pos <= 180; pos += 1) {
      myservo.write(pos);
      delay(15);
   }

   //varia la posicion de 0 a 180, con esperas de 15ms
   for (pos = 180; pos >= 0; pos -= 1) {
      myservo.write(pos);
      delay(15);
   }
}
```

SERVOMOTORES. EJERCICIO 2
CONTROL DE UN SERVOMOTOR A TRAVÉS DE UN POTENCIÓMETRO



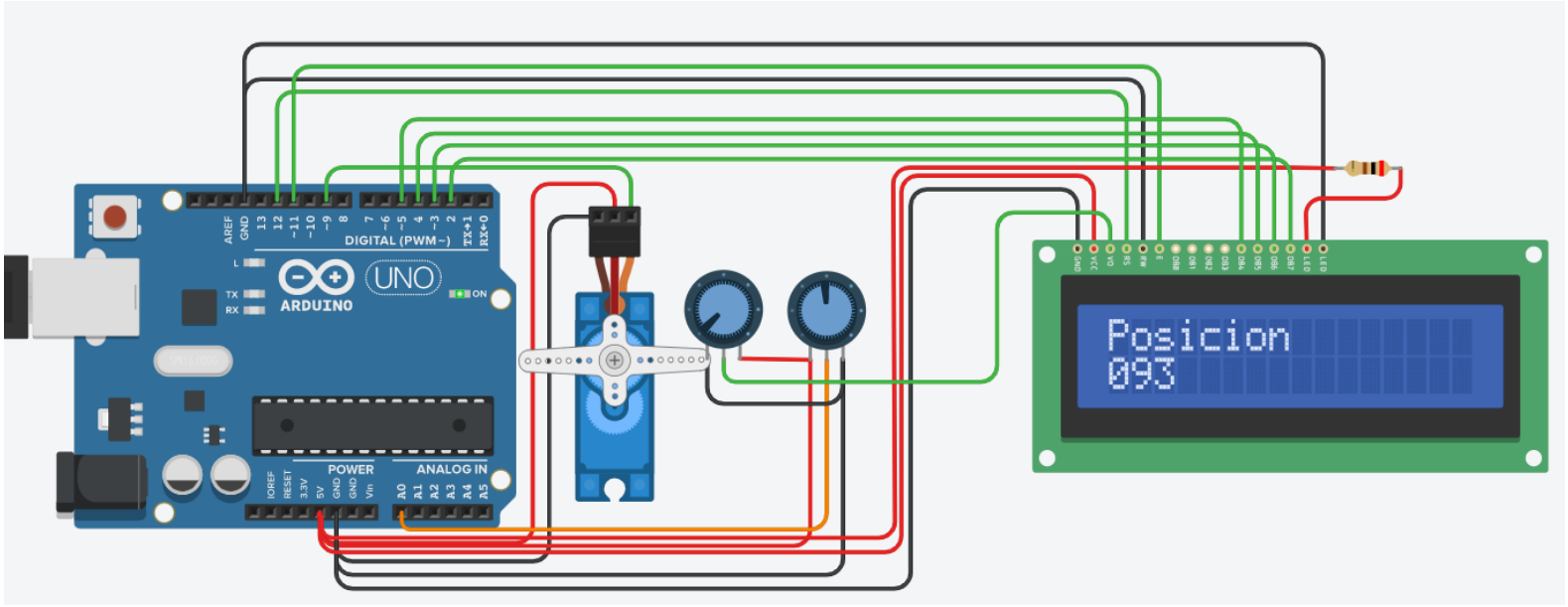
SERVOMOTORES. EJERCICIO 2
CONTROL DE UN SERVOMOTOR A TRAVÉS DE UN POTENCIÓMETRO

```
#include <Servo.h>

Servo myservo;

void setup(){
  myservo.attach(9);
   Serial.begin(9600);
}

void loop() {
  int adc = analogRead(A0);
  int angulo = map(adc, 0, 1023, 0, 180);
  myservo.write(angulo);
  Serial.print("Angulo:  ");
  Serial.println(angulo);
  delay(10);
}
```



EJERCICIO 3: UN POT + 2 SERVO

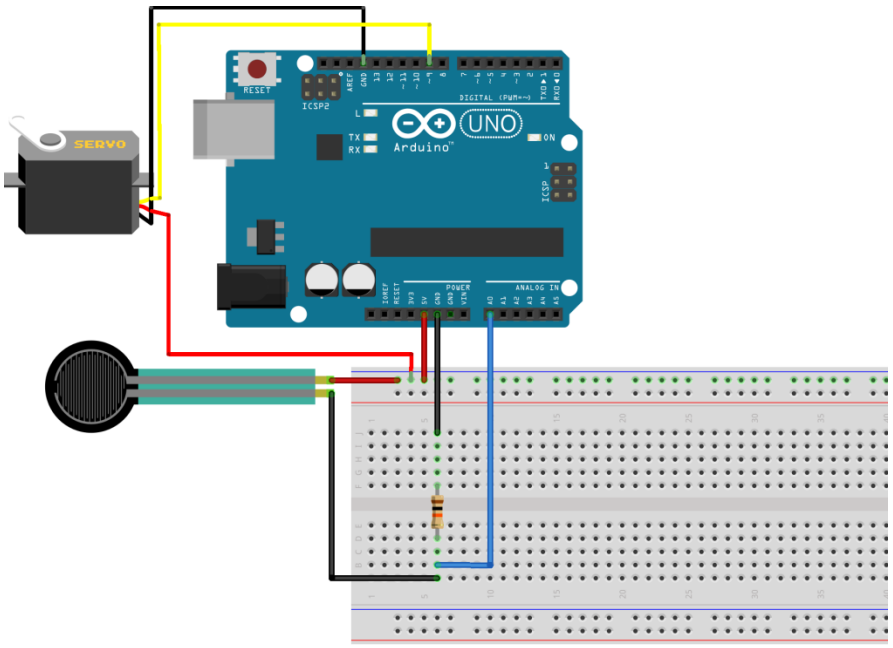
EJ. 4 SERVOMOTOR POR POTENCIÓMETRO Y LECTURA EN LCD




```ino
#include <Servo.h>
#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
Servo miServo;
int pot_pin = 0;
int servo_pin = 9;
int val;
int val2;

void setup() {
  lcd.begin(16, 2);
  miServo.attach(servo_pin);
}
```

```ino
void loop() {
  lcd.setCursor(0, 0);
  lcd.print("Posicion");

  val = analogRead(pot_pin);
  val2 = map(val, 0, 1023, 0, 179);
  miServo.write(val2);
  delay(15);

  if (val2 < 100) {
    lcd.setCursor(0, 1);
    lcd.write("0");
    lcd.setCursor(1, 1);
    lcd.print(val2);
  } else {
    lcd.setCursor(0, 1);
    lcd.print(val2);
  }
}
```

EJ. 5 CONTROL DE UN SERVO CON UN SENSOR RESISTIVO DE FUERZA




```
#include <Servo.h>

Servo servo;
int reading;

void setup(void) {
  Serial.begin(9600);
  servo.attach(9);
  servo.write(0);
}

void loop(void) {
  reading = analogRead(A0);
  Serial.print("Sensor value = ");
  Serial.println(reading);

  int value = map(reading, 0, 1023, 0, 255);

  servo.write(value);
  delay(10);
}
```